

Northern University, Nowshera

Application of Information and Communication Technologies (AICT)

(Week 5) Lecture 9 & 10

Objectives: Learning objectives of this lecture are

- **Computer Languages**
- **Assembly & High-level Languages**
- **Compiler, Linker & Interpreter**

Text Book & Resources: Introduction to Computers 6th International Edition, Peter, N. McGraw-Hill

Computer Languages

A programmer uses computer languages to instruct a computer what he/she wants to do. All computer languages have been classified broadly into three categories:

1. Machine Language
2. Assembly Language
3. High-Level Language

Machine Language:

- Only language of a computer understood by it without using a translation program.
- Normally written as string of binary 1s and 0s.
- Written using decimal digits if the circuitry of the computer being used permits this.
- Typical Machine Language Instruction format is

OPCODE (operation code)	OPERAND (Address/Location)
----------------------------	-------------------------------

OPCODE: Tells the computer which operation to perform from the instruction set of the computer.

OPERAND: Tells the address of the data on which the operation is to be performed.

Northern University, Nowshera

Application of Information and Communication Technologies (AICT)

Characteristics:

- Can be executed very fast (because no need for further translation).
- Machine Dependent (as internal design of every computer is different so, machine language also vary computer to computer)
- Difficult to Program (Programmer need to remember OPCODE and storage location of OPERAND)
- Error Prone (Lack of indication for error)
- Difficult to Modify

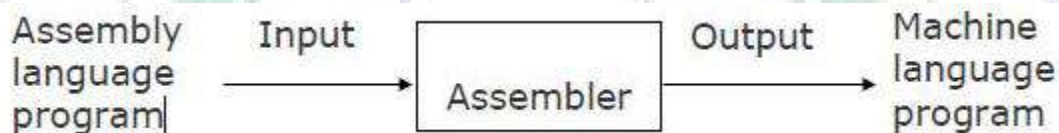
Assembly Language:

A language that allows use of letters and symbols instead of numbers for representing instruction and storage location is called as assembly language.

Unlike machine language, uses ADD instead of binary 1110 or decimal 14.

Allow storage location to be represented in form of alphanumeric addresses instead of numeric addresses e.g. represents memory location 1000, 1001, and 1002 as FRST, SCND, and ANSR respectively.

As computer can only execute machine language directly, so an assembly language code needs to be translated into machine language. A translator program, that perform assembly language to machine language is known as, Assembler.



Advantages:

- Easier to understand and use (as it uses symbols and letters to represents OPCODE and addresses).
- Easier to locate and correct error (with the help of assembler).
- Easier to Modify.
- No worry about sequence number of addresses.
- Efficiency of machine language (As there is one-to-one correspondence between assembly language and machine language, so it enjoys the efficiency of its corresponding machine language).

Northern University, Nowshera

Application of Information and Communication Technologies (AICT)

Limitation:

- Machine dependent (Different Assemblers for different Computers)
- As Machine dependent so required knowledge of Hardware
- Machine level coding (A there is one-to-one correspondence between machine and assembly language, so there will be as many instructions in assembly as we will have in machine language)

High Level Language:

- Machine Independent
- Do not require programmers to know anything about the internal structure of computer on which high-level language programs will be executed
- Enable programmers to write instructions using English words and familiar mathematical symbols and expressions

Advantages:

- Machine independent
- Easier to learn
- Fewer errors
- Better Documentation

Limitations:

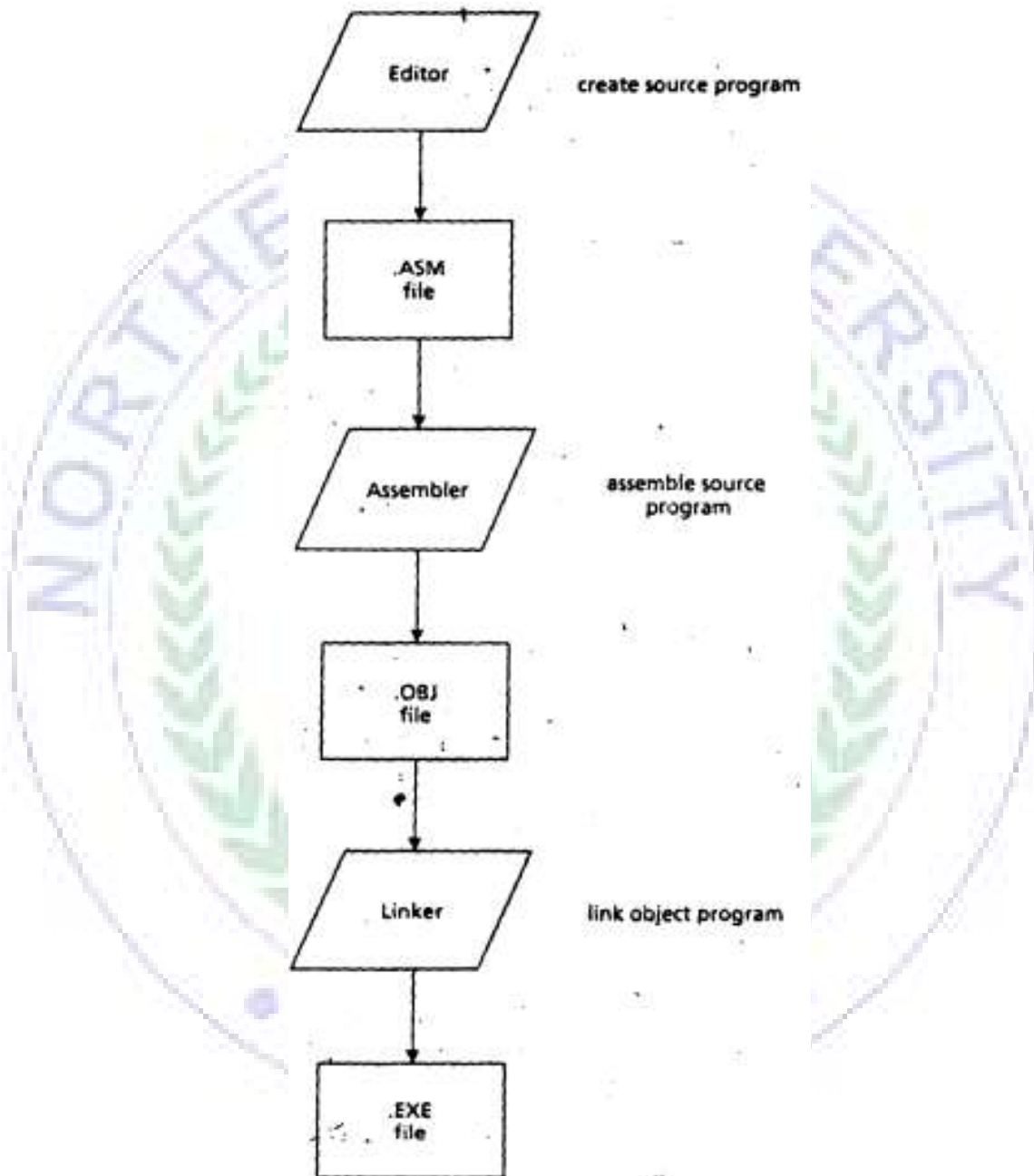
- Lower efficiency
- Less flexibility

How Program executes?

- User writes a program in C language (high-level language) using IDE.
- The C compiler, compiles the program and translates it to assembly program (low-level language).
- An assembler then translates the assembly program into machine code (object).
- A linker tool is used to link all the parts of the program together for execution (executable machine code).
- A loader loads all of them into memory and then the program is executed.

Northern University, Nowshera

Application of Information and Communication Technologies (AICT)



Video Link:

<https://www.youtube.com/playlist?list=PLEby327DX1cj2K1JNdj18XGwACKF4MwYm>

Northern University, Nowshera

Application of Information and Communication Technologies (AICT)

Compiler:

A compiler is a special program that processes statements written in a particular programming language and turns them into machine language or "code" that a computer's processor uses.

Interpreter:

As the name suggests, an interpreter transforms or interprets a high-level programming code into code that can be understood by the machine (machine code) or into an intermediate language that can be easily executed as well.

The interpreter reads each statement of code and then converts or executes it directly.

Compiler VS Interpreter:

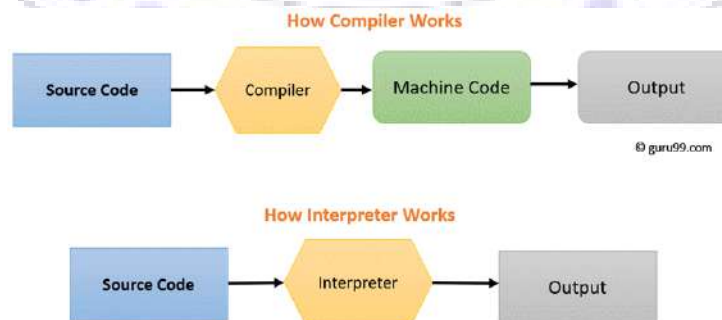
Humans can only understand high-level languages, which are called source code. Computers, on the other hand, can only understand programs written in binary languages, so either an interpreter or compiler is required.

Programming languages are implemented in two ways: interpretation and compilation.

The interpreter reads each statement of code and then converts or executes it directly. In contrast, an assembler or a compiler converts a high-level source code into native (compiled) code that can be executed directly by the operating system (e.g. by creating a .exe program).

Compilers must generate intermediate object code that requires more memory to be linked, contrarily to interpreters which tend to use memory more efficiently.

Programming languages that use interpreters include Python, Ruby, and JavaScript, while programming languages that use compilers include Java, C++, and C.



Northern University, Nowshera

Application of Information and Communication Technologies (AICT)



Northern University, Nowshera

Application of Information and Communication Technologies (AICT)

Linker:

In computer science, a linker is a computer program that takes one or more object files generated by a compiler and combines them into one, executable program.

Computer programs are usually made up of multiple modules that span separate object files, each being a compiled computer program. The program as a whole refers to these separately compiled object files using symbols. The linker combines these separate files into a single, unified program, resolving the symbolic references as it goes along.

